PD3

⑲日本国特許庁　　⑪特許出願公開

公開特許公報　　昭53—110338

⑤1Int. Cl.²	識別記号	⑤2日本分類	庁内整理番号	④3公開 昭和53年(1978)9月27日
G 06 F 5/02		97(7) E 21	7165—56	

発明の数 1
審査請求 有

(全 18 頁)

⑤4符号化装置

⑦1特　　願　昭53—23698
⑦2出　　願　昭53(1978)3月3日
優先権主張　③2 1977年3月4日 ③3 アメリカ国(US) ③1 774365

⑦2発 明 者　グレン・ジョージ・ラングドン・ジュニア
アメリカ合衆国カリフオルニア州サン・ホセ・ハンプトン・ドライブ6672番地

⑦2発 明 者　ジョーマ・ジョーハネン・リザネン
アメリカ合衆国カリフオルニア州サン・ホセ・アドニス・ウエイ2442番地

⑦1出 願 人　インターナシヨナル・ビジネス・マシーンズ・コーポレーシヨン
アメリカ合衆国10504ニューヨーク州アーモンク(番地なし)

⑦4復代理人　弁理士　頓宮孝一

明　細　書

1. 発明の名称　符号化装置

2. 特許請求の範囲

(1) 任意の順序で番号付けられたN個のシンボル a_1 乃至 a_N を含む情報源から送られてきたシンボル列を符号化する際に、該シンボル列中の各シンボル毎に順次符号化操作を行なつて、その度に新しい符号列を生成する符号化装置にして、

上記シンボル列中の各シンボル a_k に応答して、その生起確率 $p(a_k)$ によつて決まる長さ表示 $\ell(a_k)$ を引出すための第1メモリ手段と、

上記長さ表示 $\ell(a_k)$ から、新しい符号列の長さ表示Lの整数部分y及び小数部分xを計算するための演算手段と、

上記小数部分xと、上記情報源におけるシンボルの番号付けに従う累積生起確率 P_{k-1} とによつて決まるrデイジットの値 A_k を発生するための第2メモリ手段と、

先行符号列の上位rデイジットのうちの下位のyデイジットをそのまま出力するためのシフト手段と、

先行符号列の上位のr－yデイジットと上記値 A_k とを加算して新しい符号列の上位rデイジットを生成するための加算手段とを有する符号化装置。

(2) 上記長さ表示 $\ell(a_k)$ の値は、$\sum_{k=1}^{N} 2^{-\ell(a_k)} \leq 1$ によつて制限される特許請求の範囲第(1)項記載の符号化装置。

(3) 上記演算手段は、各符号化操作において先行符号列の長さ表示L′と上記長さ表示 $\ell(a_k)$ との和から上記整数部分y及び上記小数部分xを得る特許請求の範囲第(2)項記載の符号化装置。

(4) 上記演算手段は、各符号化操作において、先行符号列の長さ表示L′の小数部分x′と上記長さ表示 $\ell(a_k)$ との和から上記整数部分y及び上記小数部分xを得る特許請求の範囲第(2)項記載の符号化装置。

(5) 上記第2メモリ手段によつて発生される値 A_k は、$P_{k-1} \cdot 2^x$ をrデイジットで近似したものである特許請求の範囲第(1)項、第(2)項、第(3)項

又は第(4)項記載の符号化装置。但し、$P_{k-1}=\sum_{i=1}^{k-1}p(a_i)$、$P_0=0$、$P_N=1$である。

(6) 上記シフト手段は、先行符号列の上位rデイジットをyデイジット分だけ右シフトさせる右シフト装置から成つている特許請求の範囲第(5)項記載の符号化装置。

(7) 上記第1メモリ手段及び上記第2メモリ手段は、上記情報源に含まれるすべてのシンボルについて予め計算された$\ell(a_k)$及びA_k（$k=1$、2、・・・、N）を記憶している特許請求の範囲第(1)項乃至第(6)項の1つに記載の符号化装置。

3. 発明の詳細な説明

本発明は、有限の情報源アルファベットからのシンボル列の符号化（短縮（compaction）を行なうため、シンボルの相対生起頻度を利用するもの）及び復号に関するものであり、特に可算（enumerative）符号化法及びハフマン短縮法の改良に関するものである。

ハフマン短縮法（符号化法）は情報源アルファベットの各文字に符号ワードを割当てる際に、この符号ワードの長さを文字の相対生起頻度に反比例させるものである。生起統計は、静的又は準静的であつた。即ち、この方法は、一定の適応性ある符号ワードの再配列及び再割当てを考慮していなかつた。

1952年9月に発行されたProceedings of the IREの1098乃至1101頁に記載されているハフマンの最初の論文"最小冗長符号の構成方法（A Method for the Construction of Minimum Redundancy Codes）"以来、例えば米国特許第3675211号明細書、同第3675212号明細書、同第3701111号明細書、同第3694813号明細書及び同第3717851号明細書に示されているように、幾多の修正がなされてきた。このような符号化法のバックグランドについては、1963年にマグロー・ヒル社から発行されたノーマン・アブラムソン著の"情報論理及び符号化（Information Theory and Coding）"の第3章及び第4章を参照されたい。しかしながら、これは、符号ワードが先験シンボルの固定シーケンスから構成され、すべてのワードが区別でき（非特異性）、そしてどの符号ワードも別の符号ワードのプレフイクスではない（一意復号可能）という古典符号の域を出ない。

可算符号化法は、例えばn個のシンボルから成る2進列の各置換を符号化するもので、まず2^n個の項を順番に並べて、各項を序数で表示することにより行なわれる。可算符号化法の詳細については、1973年1月に発行されたIEEE Transactions on Information Theoryの73乃至77頁に掲載されているコーバー（T. M. Cover）の論文"可算情報源符号化法（Enumerative Source Encoding）"を参照されたい。簡単に説明すると、例えばn個の2進デイジットのうち1がm個ある2進列の集合は、T（n、m）で表わされる。項は、一貫した方法で、即ち予め決めた優先順位（左から右）に基づいて大きさによつて（辞書式順序）順番に並べる必要がある。例えば、$n=3$及び$m=2$とすると、011＜101＜110のように順序付けてもよい。順序付けにおいて各項に位置番号を割当てると、最終項は2項係数$\binom{n}{m}=n!/(n-m)!m!$になる。他のすべての位置番号（符号ワード）はこれよりも小さい。

ハフマン符号化法及び可算符号化法においては、シンボル当りの達成可能な平均符号長は、情報源シンボルの頻度についてのエントロピー関数によつて決まる下限を有している。

情報源シンボル列sを$s=a_i a_j \cdots$（右側に延びる）で表わし、これの終端（右端）に新しいシンボルa_k（$k=1$、2、・・・、N）を付加することによつて得られる新しいシンボル列をsa_kで表わすものとする。ハフマン符号化法においてはsa_kの符号化表示は、sの符号の右端にシンボルa_kの符号ワードA_kを付加即ち連結することによつて得られる。これは、符号ワード長がシンボルの生起頻度とは無関係に付加シンボルのビット数だけ長くなることを意味する。言い換えれば、ハフマンの符号化は、2進情報源の列に

対しては短縮を行ない得ず、また少数のアルファベットに対しては不十分な短縮しか行なわない。

ハフマンの短縮法を改良するため、少数の情報源アルファベットに対して、ブロック化として知られている技法が提案された。ブロック化においては、新しい情報源アルファベットの新しい情報源シンボルは、元の2以上の情報源シンボルの並置（juxtaposition）である。一度にk個のシンボルをまとめてとりあげると、情報源アルファベットの大きさはN^kまで増大する。

可算符号化は、このようなブロック化を必要としないが、新しい各符号ワード$C(sa_k)$の計算には、元の符号ワード即ち符号列C(s)の全ビットについての情報が必要となる。従つて、限りなく増大する記憶容量を有するメモリが必要である。

本発明の目的は、ハフマン法におけるブロック化及び単純可算符号化を支持するための無限大の（限りなく増大する）メモリを必要とすることなく、符号列特に2進ビット列の符号化及び復号を行なうための装置を提供するにある。

本発明の他の目的は、シンボルの並べ方及び情報源アルファベットの大きさには実質的に無関係な算術列符号化法を用いる装置を提供するにある。

これらの目的は、情報源シンボルの相対生起頻度が既知であるか又は推定可能であれば、達成することができる。

シンボルの列$s=a_i a_j \cdots$を考えると、ハフマンの符号化によれば、有限の情報源アルファベットから取り出される一意的な各シンボルに対し、先験符号ワードが割当てられる。可算符号化においては、列sは符号化された列C(s)として表わされ、列sに次のシンボルa_kが付加されると、新しい符号列$C(sa_k)$の生成には、先行符号列C(s)の全体を考慮する必要がある。

新しい符号列$C(sa_k)$は、情報源アルファベットのすべてのシンボルを任意に順序付けた場合の各シンボルa_kの順序数即ち位置番号kと、長さ表示の小数部分及び整数部分と、先行符号列C(s)の最後の数デイジットとで表わされ得ることが判明した。この事実から、本発明においては、まず長さ表示が決定され、次いでその小数部分及び整数部分を一意的に利用することにより新しい符号列が生成される。符号化を行なう場合には、新しい符号列のための長さ表示$L(sa_k)$が決定された後、新しい符号列が計算される。これは、次の帰納的数式（recursion）によつて簡単に表わされる。

$$L(sa_k)=L(s)+\ell(a_k) \quad (1)$$

$$C(sa_k)=C(s)+P_{k-1}\cdot 2^{L(sa_k)} \quad (2)$$

上式において、L(s)は先行符号列C(s)のための長さ表示であり、$\ell(a_k)$は$p(a_k)$を事前生起確率としたときの$\log_2\{1/p(a_k)\}$を有理数で近似したものである。シンボルa_kに対応するこの長さ表示$\ell(a_k)$はクラフトの不等式：

$$\sum_{k=1}^{N} 2^{-\ell(a_k)} \leq 1 \qquad (\text{Nはシンボル数})$$

を満足するものでなければならない。P_{k-1}は情報源アルファベットにおけるシンボルの任意の順序付けに基づいた累積生起確率であつて、次式で表わされる。

$$P_{k-1}=\sum_{i=1}^{k-1} p(a_i),\ P_0=0,\ P_N=1.$$

既に述べたように、本発明においては、長さ表示の小数部分及び整数部分が共に利用される。例えば、

$$\begin{aligned} L(sa_k) &= L(s)+\ell(a_k) \\ &= \text{整数}\ y+\text{小数}\ x \\ &= y(sa_k)+x(sa_k) \end{aligned}$$

とおいて、これを(2)式に代入すると、次のようになる。

$$\begin{aligned} C(sa_k) &= C(s)+P_{k-1}\cdot 2^{L(sa_k)} \\ &= C(s)+P_{k-1}\{2^{y(sa_k)}\cdot 2^{x(sa_k)}\} \end{aligned} \quad (3)$$

yは整数であるから、レジスタにおいて$2^{\pm y}$を乗算することは、$+y$については左シフトを、$-y$については右シフトを各々行なうことになる。因数2^xは、qビットで近似（$e(x)\simeq 2^x$）することができる。これはテーブル索引（table

look up）に役立つ。

これらの考察から、本発明の一実施例においては、新しい符号列は次のようにして生成される。

(1) $\ell(a_k)$についてのテーブル索引
(2) $L(sa_k)$を得るための加算
(3) $e(x) \simeq 2^x$についてのテーブル索引
(4) P_{k-1}及び$e(x)$の乗算
(5) yビットの左シフト
(6) $C(s)$及び$P_{k-1}e(x)2^y$の加算

本発明の別の実施例では、ステップ(3)及び(4)が組合わされ、そしてステップ(5)では右シフトが行なわれる。これは、(3)式を次のように書き直したものに基づいている。

$$2^{-y(sa_k)}C(sa_k)=2^{-y(sa_k)}C(s)+A_k$$

但し、

$$A_k=P_{k-1}2^{x(sa_k)}$$

である。

上式によれば、先行符号列$C(s)$を$y(sa_k)$ビットだけ右シフトしてA_kに加算すると、同じく$y(sa_k)$ビットだけ右シフトされた符号列$C(sa_k)$が得られる。ここで、$y(sa_k)$が新しい符号列$C(sa_k)$の長さ表示$L(sa_k)$の整数部分、即ち、（先行符号列$C(s)$の長さ表示$L(s)$＋付加シンボルa_kの長さ表示$\ell(a_k)$）の整数部分であることを考えると、上式の右辺においてC(s)をy(sa_k)ビットだけ右シフトさせることは、前回の計算で得られた$2^{-y(s)}C(s)$を、$L(s)$の小数成分$x(s)$と$\ell(a_k)$との和の整数成分の値だけ右シフトさせることと同等である。従つて、$y(sa_k)$として$x(s)+\ell(a_k)$の整数成分の値を用い、上式の左辺を新しく$C(sa_k)$とおくと、次式が得られる。

$$C(sa_k)=2^{-y(sa_k)}C(s)+A_k \qquad (4)$$

上式に従う符号化手順は次の通りである。

(1) $\ell(a_k)$についてのテーブル索引
(2) $L(sa_k)=y(sa_k)+x(sa_k)$を得るために、$x(s)$と$\ell(a_k)$とを加算
(3) A_kについてのテーブル索引
(4) $y(sa_k)$ビットだけ$C(s)$を右シフト
(5) A_k及び$2^{-y(sa_k)}\cdot C(s)$を加算

この手順によれば、新しい符号列の計算が簡単になり、シフトされた先行の符号列にA_kを加算するだけでよい。これはA_kが位置番号k及び小数$x(sa_k)$の関数であるという事実に基づいている。更にこの手順は、A_k及び$C(s)$が2進小数点を基準にして同じ相対位置を保持するということを利用している。この点について、P_{k-1}及び$x(sa_k)$が0と1の間にあることを想起されたい。

符号化においては、情報源シンボル列は符号化装置へ直列に入力され、符号化された2進列がそこから発生される。この符号化2進列は、その開始点及び終点を考慮することによつてのみ復号され得る。もし$C(sa_k)$が0と2の間の大きさを有するものに制限されるならば、$C(sa_k)$の2進小数点は、最後の被加数A_kと桁列され得る。従つて、$C(sa_k)$の左端（先頭）のビットが整数部分を表わし、その右側のビットが小数部分を表わす。

符号化装置の内部では、符号列は、r個のデイジットから成つている。この"r"デイジットは、A_kについての精度のインデックスでもある。C(s)の下位のyデイジットは、1サイクルの間に符号化装置から桁送りによつて出力され、上位デイジットのr－yデイジットが新しい計算において列に付加される。パラメータ"r"は、生起頻度の最も低いシンボルの生起確率をp_mとしたときに、$\log(1/p_m)$以上の値に設定されるのが好ましい。

復号においては、圧縮された（compressed）ビット列を復号可能な文字へ"解析（parse）"し、次いで符号化のときと同じような方法で文字を列から"分離（detach）"するために、長さ表示を使用する必要がある。具体的に言うと、も

し復号装置が2^{r+q}個のワードを記憶し得るメモリを有していると（rはA_kのビット長）、復号には次のような手順が必要になる。

(1) 符号化された列C（sa_k）の開始点の位置決め。

(2) C（sa_k）の次のrビット及び保持されている小数部分x（sa_k）のqビットによるメモリのアドレス指定、並びにa_k及び$\ell(a_k)$の検索。

(3) 復号装置を圧縮された符号列の次の部分と整列させるために、符号列をy（sa_k）ビットだけ左シフト。

本発明に従う短縮法は、ブロック化が不要なだけでなく、有限のメモリしか必要とせず、更にエラーの制御及び符号ワード・テーブルのためのメモリ容量の制御も可能である。

例えば、2進アルファベットのハフマン符号化法においては、エントロピー即ち符号化されたシンボル当りの理論上の最小平均長をH(s)、符号化されたシンボル当りの平均長をLn/nとしたときに、$H(s) \leq Ln/n < H(s)+1/n$なる不等式（シャノンの第1定理参照）を満足させるためには、nビットのブロック化が必要である。ハフマン符号ワードのためのテーブルは、2^n個の符号ワードを記憶し得る容量を有している必要があり、また多くの符号ワードは、nよりも多いビット数を有する。

ところが、算術符号化法においては、同じ上限$H(s)+1/n$に対し、大きさがnのオーダーであるテーブルを用いることができる。この場合の各ワードの長さは、$\log_2 n$のオーダーである。

また、算術符号化法は、符号化文字の短縮の他にも、現在の希薄行列（sparse matrix）法とは別の、非符号化データ（ラスタ符号化データ）の列に対する圧縮法をも提供する。ここで重要なことは、符号化データ及び非符号化データに対して別々の手法乃至は機構を用いるものとは異なり、2進列符号化のための単一の短縮法が存在するであろうということである。

以下、図面を参照しながら、本発明の実施例について説明する。

第1A図は、記憶及び伝送システムと共に用いられる2進列符号化装置の概略を示したものである。情報源1は、情報源アルファベット$\{a_1, a_2, \cdots a_k, \cdots a_N\}$から選択された一連のシンボル$a_k$を出力する。これらのシンボルは、バス3を介してバッファ／コントローラ5へ直列に供給され、そこに累積される。このバッファ5は、情報源1と伝送の前に必要な信号の条件付け又は符号変換との間における必要な速度整合を行なう。記憶／伝送システムは、例えば、利用装置29を駆動する後入れ先出し（LIFO）記憶ユニット15を含んでいる。通常、速度整合及び符号変換は、利用装置の側においても必要であるから、記憶ユニット15と利用装置29との間にバス17及び27を介して別のバッファ／コントローラ19が接続される。

バッファ5に累積された情報源1からのシンボルは、次にバス7を介して符号化装置9の如き変換装置へ送られる。符号化装置9は1対1の写像（変換）を実行した後、バス11を介して変換されたシンボルをバッファ5へ送る。刻時動作及び各符号文字の長さに関する制御情報は、別の並列バス12を介して送られる。同様に、記憶ユニット15から受取られてバッファ19に累積された符号化文字の変換は、復号装置23で行なわれる。復号装置23は、情報源1から発生された元のシンボルを再生するものであるが、バッファ19からバス21を介して供給された2進列を正しく解析するためには、元の文字及び長さ情報を引出さねばならない。長さ情報はバス22を介して送られ、情報源文字データはバス25を介して送られる。

ここで、第1B図及び第1C図を参照しながら、従来のハフマン法に従う符号化及び復号について説明しておく。ハフマン符号は、各情報源シンボルの事前生起確率$p(a_k)$（$k=1, 2, \cdots, N$）を利用して、可変長$\ell(a_k)$の対応する符号ワードを割当てたものである。この場合、$p(a_k)$

及び$\ell(a_k)$については、$p(a_k) \geq 2^{-\ell(a_k)}$及び$\sum_{k=1}^{N} 2^{-\ell(a_k)} \leq 1$が成立している。符号化列が"0"及び"1"の組合わせから成つていると、$\ell(a_k)$は、シンボルa_kに対するハフマン符号ワードのビット数に相当する。

ハフマン符号化法においては、情報源シンボル列が受取られると、各情報源シンボルa_kに応答してメモリから符号ワードA_k及び長さ表示$\ell(a_k)$が検索され、次いで、前に符号化された列Cが$\ell(a_k)$ビットだけ右又は左にシフトされた後、符号ワードA_kが付加される。従つて、情報源シンボル列$a_i a_j a_k$をハフマン法で符号化した場合の符号列は、$A_i A_j A_k$で表わされる。これは、符号ワードの連結と呼ばれている。

第1B図に示したハフマン符号化装置においては、第1A図のバッファ5からバス7を介して供給された情報源シンボルa_i、a_j、a_kは一旦入力レジスタ31に保持される。メモリ・アドレスa_kによつてアドレス指定されるメモリ33の特定の記憶位置には、対応するハフマン符号ワードを記憶し得るメモリに対してはメモリ47をアドレス指定するために、最初の$\log_2(1/pm)$ビットが入力レジスタ43へ送られる。メモリ47から読出される情報は、情報源シンボルa_k及びその長さ$\ell(a_k)$である。$\ell(a_k)$はレジスタ49へロードされ、a_kは出力レジスタ51へロードされる。レジスタ49の内容は、次の符号ワードの復号のために新しい開始点を整列させるようにレジスタ43をシフトさせるため、シフト制御装置45へ送られる。

ハフマン型の圧縮符号ワードの長さは、整数値に限定されることは明らかである。符号化における一般的な原則として、各符号ワードの長さ$\ell(a_k)$と、情報源アルファベットにおける対応するシンボルa_kの相対生起頻度$p(a_k)$とは、次のような関係を有している。

$$p(a_k) \geq 2^{-\ell(a_k)}$$

これを書き直すと、

$$p(a_k) = 2^{-\ell(a_k)} + E$$

ドA$_k$及び長さ表示$\ell(a_k)$が記憶されている。メモリ33から読出された符号ワードA$_k$はレジスタ35へロードされ、長さ表示$\ell(a_k)$はレジスタ37へロードされる。

レジスタ35へロードされたA$_k$は、次に出力シフト・レジスタ41の所定の位置へ送られ、一方、レジスタ37へロードされた$\ell(a_k)$は、シフト制御装置39へ送られる。シフト制御装置39は、次の符号ワードを出力レジスタ41へロードし得るようにすると共に転送バス11へ符号ワードを出力するため、符号ワードA$_k$を所定の位置数だけシフトさせる。

第1C図のハフマン復号装置は、米国特許第3883847号明細書に示されているような型のものである。バッファ/コントローラ19からの圧縮された符号ワード列は、バス21を介して入力レジスタ43へロードされる。最大符号ワード長は先験的に知られている。従つて、最長の単位符号ワードの長さ(ビット数)を$\log_2(1/pm)$としたときに、$\log_2(1/pm)$個のワードを記憶し得るメモリに対してはメモリ47をアドレス指定するために、最初の$\log_2(1/pm)$ビットが入力レジスタ43へ送られる。メモリ47から読出される情報は、情報源シンボルa_k及びその長さ$\ell(a_k)$である。$\ell(a_k)$はレジスタ49へロードされ、a_kは出力レジスタ51へロードされる。レジスタ49の内容は、次の符号ワードの復号のために新しい開始点を整列させるようにレジスタ43をシフトさせるため、シフト制御装置45へ送られる。

ハフマン型の圧縮符号ワードの長さは、整数値に限定されることは明らかである。符号化における一般的な原則として、各符号ワードの長さ$\ell(a_k)$と、情報源アルファベットにおける対応するシンボルa_kの相対生起頻度$p(a_k)$とは、次のような関係を有している。

$$p(a_k) \geq 2^{-\ell(a_k)}$$

これを書き直すと、

$$p(a_k) = 2^{-\ell(a_k)} + E$$

となり、従つて次式が得られる。

$$\ell(a_k) = \log_2\{1/p(a_k)\} + E'$$

これから明らかなように、もし$\ell(a_k)$が整数であれば、$p(a_k)$は無理数になる。従つて、符号ワード長が整数値をとる圧縮符号化系の圧縮効率は、符号ワード長に小数部分が含まれるような符号化系の効率よりも劣つている。可算符号及び算術列符号には、このような小数部分が含まれる。

本発明に従う符号化アルゴリズム

情報源シンボルa_i及びa_jから成るシンボル列$s = a_i a_j$を考えると、これに新しいシンボルa_kを付加した場合には、シンボル列は$sa_k = a_i a_j a_k$と表わせる。ここで、シンボル列sを符号化装置へ供給したときの出力をC(s)で表わすと、可算符号化及び列符号化においては、$s \rightarrow C(s)$及び$sa_k \rightarrow C(sa_k)$という対応関係がある。符号化は、次のようにして帰納的に定義さ

れる。

$$L(sa_k)=\text{長さ表示}=L(s)+\ell(a_k)$$

$$C(sa_k)=\text{圧縮された符号列}=C(s)+P_{k-1}\cdot 2^{L(sa_k)}$$

上式中の$\ell(a_k)$及びP_{k-1}は次式で表わされる。

$$\ell(a_k)=\log_2\{1/p(a_k)\},$$

$$P_{k-1}=\sum_{i=1}^{k-1}p(a_i);\ P_0=0;\ P_N=1$$

但し、$\ell(a_k)$には

$$\sum_{k=1}^{N}2^{-\ell(a_k)}\leq 1$$

という制限がある。

長さ表示$L(sa_k)$を、整数成分及び小数成分に分けて考えると、圧縮された符号列$C(sa_k)$の計算をより簡単に行なうことができる。即ち、

$$L(sa_k)=y(sa_k)\text{（整数）}+x(sa_k)\text{（小数）}$$

$$C(sa_k)=C(s)+P_{k-1}\cdot 2^{y(sa_k)}\cdot 2^{x(sa_k)}$$

ここで、$A_k=P_{k-1}\cdot 2^{x(sa_k)}$とおくと、

$$C(sa_k)=C(s)+A_k\cdot 2^{y(sa_k)}$$

これは次のように書き直せる。

$$2^{-y(sa_k)}\cdot C(sa_k)=2^{-y(sa_k)}\cdot C(s)+A_k$$

上式中のA_kについては、$0\leq P_{k-1}<1$及び$0\leq x(sa_k)<1$であるから、$0\leq A_k<2$が成立する。また、A_kはk及びxの関数であるから、その値は、k及びxに基づくテーブル索引を利用して得ることができる。しかしながら、符号列$C(sa_k)$の生成においては、先行符号列C(s)とテーブルから引出された値A_kとを正しく整列させることが必要である。言い換えれば、先行符号列とテーブルからの値との間の相対的な位置関係を維持しておくことが重要である。

上述の2種類の符号化手順のうちの後者を実施するための算術列符号化装置の一例を第2図に示す。図示の符号化装置は、バス7上の情報源シンボルa_kに応答して、バス11へ圧縮された符号列$C(sa_k)$を供給し、更にバス12へ制御情報を供給する。この制御情報には、線69へ供給される被保持小数成分及び線77へ供給されるシフト量が含まれる。

タイミング

第2図の符号化装置及び第3図の復号装置のためのタイミング方式は、同期式単相型の簡単なもので、回路遅延、フリップフロップの設定に要する時間及び伝播遅延が最悪の場合にもうまく動作させるため、刻時パルス間のパルス間隔は十分に長くとられている。これについては、1972年にマグロー・ヒル社から発行されたジエイ・ビー・ビートマン著の"デイジタル・システムの設計（The Design of Digital Systems）"の161乃至170頁を参照されたい。第4A図は、符号化装置における最悪の場合の各遅延の様子を刻時パルスCPと共に示したもので、第4B図は同じく復号装置におけるものである。刻時スキューは、最大許容公差の範囲内にあるものとする（上述のテキストの167乃至170頁参照）。

第2図に示した符号化用メモリ57及び59並びに第3図に示した復号用メモリ99、109及び117には、初期設定段階の間に必要な値がロードされる。これらのメモリは、読出し／書込みメモリ、読出し専用メモリ及びプログラム可能な読出し専用メモリの何れであつてもよい。このようなメモリをロードするための手段については周知であるから、ここでは特に述べない。Cレジスタ87は、前に符号化された列の上位rビット即ち算術オペランド部分を保持し、Xレジスタ67は、qビットの被保持小数成分を保持する。これらのレジスタは、普通"0"に初期設定されるが、Xレジスタ67には、任意の値を初期設定してもよい。Cレジスタ87の初期設定値は、あとで説明する値$max\,C_x$以下になるように制限される。可算符号化においては、先行符号列C(s)の算術オ

ペランド部分は、この列C(s)の全体であるが、本発明では、Cレジスタ87は、先行符号列C(s)の算術オペランドを構成し得る上位のrビットのみから成る被保持列セグメントを保持する。残りの下位ビットはすべてシフトされるだけである。Cレジスタ87の内容は、算術的使用の前に、$y(sa_k)$ビットだけ右方向にシフトされる。この結果、Cレジスタ87の右端の$y(sa_k)$ビット（出力列セグメントと呼ばれる）は、バス11を介して第1A図に示したバッファ／コントローラ5へ送られ、このとき、シフト量$y(sa_k)$を表わす制御信号もバス12を介して送られる。この制御信号は、バッファ／コントローラ5に受取られるビットの数を表わす。

各情報源シンボルa_kは、入力レジスタ31に受取られた後、変換器53へ送られる。変換器53は、シンボルa_kをその位置番号に対応する数値k（nビット）へ変換する。この数値kは、バス55を介してメモリ57へ供給される。メモリ57は、$\ell(a_k)=\log_2\{1/p(a_k)\}=y(a_k)+x(a_k)$で表わされる長さ$\ell(a_k)$を含んでいる。ここで、整数成分$y(a_k)$はtビットであり、小数成分$x(a_k)$はqビットである。

第2図及び第3図に示した添字付きの短い斜線は、各々のバスが添字の数だけの平行導体から成つていることを表わしている。例えば、バス55はn本の導体を含み、またバス61及び71は各々q本及びt本の導体を含んでいる。また、丸印で囲んだ大文字のアルファベットは、同じアルファベット同志が接続されることを表わしている。

整数値$y(sa_k)$は、加算器63及び+1加算器75から得られる。小数値$x(sa_k)$は、加算器63及びXレジスタ67から得られる。$y(sa_k)$の値は、先行符号列C(s)とバス89上のA_kとを整列させて、これらを加算器83で加算することによつてC(sa_k)の上位のrデイジットを得る際に、右シフト装置79でシフトされるC(s)のデイジットの数を表わす。

小数値$x(sa_k)$は次の刻時サイクルの開始時にXレジスタ67へロードされることになるが、現サイクルの開始時点においては、前の小数値x(s)がXレジスタ67にロードされている。$x(sa_k)$は、テーブル索引によつてメモリ59から被加数A_kをアクセスするために、kと共に用いられる。メモリ59から取出されたrビットのA_kは、バス89を介して加算器83へ送られる。新しい符号化列C(sa_k)は、A_kと右シフト装置79で適切にシフトされた後のC(s)とを加算器83で加算することによつて得られる。ここで注意すべきは、先行符号化列C(s)のrビットの被保持列セグメントのうちの下位の$y(sa_k)$ビットがバッファ／コントローラ5へ送られるということである。次の刻時サイクルにおいては、符号化列C(sa_k)の新しい被保持セグメントが、バス85を介してCレジスタ87へ送られる。連続する刻時パルスCP間の時間間隔は、関連する回路の最大遅延を考慮して、$x(sa_k)$及びC(sa_k)が安定化し得るように即ち定常状態をとり得るように、設定される。各レジスタは、1973年にテキサス・インストルメンツ社から発行された"設計技術者のためのTTLデータ・ブック（The TTL Data Book for Design Engineers）"の363乃至368に記載されている74TTLファミリー、型番SN74174のような"D"型のフリップフロップで構成することができる。これに関して、バッファ／コントローラ5は、刻時パルスの発生毎に符号化装置9へ新しいシンボルa_kを供給すると共に、シフト動作によつて出力された先行符号化列C(s)の被保持列セグメントの下位のyデイジットを取り去るように構成されてもよい。最後のシンボルの符号化が終ると、バッファ／コントローラ5はC(sa_k)の最終内容及び最終被保持小数成分$x(sa_k)$を受取る。

パラメータ"r"及び"q"の大きさ

前述のように、C(sa_k)は、符号化装置9に供給されたシンボルa_kに応答して符号化された列を表わし、C(s)は先行符号列を表わしている。符号列の長さ（ビット数）は、前に符号化された文字のすべての長さ表示$\ell(a_k)$の和になつている。しかしながら、符号化装置9の内部に保持

される符号列C(s)のビットは、上位のrビットだけである。新しい長さ表示L(sa_k)が得られたときには、C(s)のr個の被保持ビットのうちの下位のy(sa_k)ビットがシフト動作によつて出力される。この結果、C(s)のr－y(sa_k)ビットが算術オペランドとして残され、A_kの下位ビットと整列される。L(sa_k)＝L(s)＋ℓ(a_k)＝y(sa_k)＋x(sa_k)であることを想起されたい。しかしながら、L(s)のうち保持されている部分は小数成分x(s)だけである。整数成分y(s)は、前の符号化処理で得られた符号列C(s)の右シフト量に対応している。従つて、前述のように、L(sa_k)はx(s)＋ℓ(a_k)＝整数y(sa_k)＋小数x(sa_k)で表わされる。

既に述べたように、符号化操作においては、A_kの値はテーブル索引によつてメモリ59から得られる。メモリ59をアクセスするためのアドレス・ビットはa_kの位置番号kを表わすnビット及び被保持小数成分x(sa_k)のqビットから成つている。この場合、qの値は、独立設計パラメータとして扱われる。

メモリ59から取出されたrビットのA_kは、C(s)のr－y(sa_k)ビットと正確に整列されねばならない。これを達成するため、C(s)はy(sa_k)ビット位置だけ右方向にシフトされる。これにより、A_k及びシフトされたC(s)の最下位ビットが正しく整列する。A_kが左シフトされた場合には、$C(sa_k)=C(s)+A_k\cdot 2^{y(sa_k)}$となり、もしC(s)が$A_k$を基準にして右シフトされると、これは上式の両辺に$2^{-y(sa_k)}$を乗じたことになる。即ち、

$$2^{-y(sa_k)}C(sa_k)=2^{-y(sa_k)}C(s)+A_k$$

となる。これが(4)式のように書き直せることは前述の通りである。

rの値は、A_kに要求される精度を決定する。設計上の重要な要素の1つに、生起頻度の最も低い情報源シンボルの生起確率pmがある。Aは累積確率Pと2^xとの積で表わされるから、

$$A_k=P_{k-1}2^{x(sa_k)}$$

$$A_{k+1}=P_k 2^{x(s'a_{k+1})}$$

において、$x(sa_k)=x(s'a_{k+1})$となる可能性を考えると、A同志を区別するためには、P同志を区別することが必要になる。従つて、A_kのビット数rは、$P_k>P_{k-1}$を満足させるため$r\geq \log(1/pm)$でなければならない。"r"を決定する際に考慮すべき他の事柄は、あとの説明から明らかとなろう。被保持小数成分x(sa_k)のビット数qは、実際の符号ワード長がどれ程理想に近いかを知るための基準になる。理想的な符号ワード長が$\log_2\{1/p(a_k)\}$であることは明らかであろう。ハフマン符号ワードの長さは整数値に限定されるので、これを上述の理想ワード長に近づけることはできない。qの値を大きくすればする程、符号ワードの長さは理想ワード長に近づく。

右シフト対左シフト

まず、情報源シンボル列におけるシンボルの位置を添字で表わして、$C_0=0$を符号列の初期値とし、$L_0=0$を長さの初期値とする。シンボル列における最初のシンボルa_1を符号化するためには、$L_1=\ell_1=y_1+x_1$及びA_1を決定する必要がある。A_1をy_1ビットだけ左シフトして、これをC_0に加算する代りに、C_0をy_1ビットだけ右シフトして、A_1に加算することも可能である。即ち、

$$C_1=A_1+C_0\cdot 2^{-y_1}$$

となる。

次に、シンボル列a_1a_2を符号化する場合には、新しい相対長さ表示は、$L_2=\ell_2+x_1$である。このステップでは、小数成分x_1のビットはシフトされずにℓ_2に加算される。L_2の整数成分をy_2で表わし、小数成分をx_2で表わすと、新しい符号列C_2は次のようになる。

$$C_2=C_1\cdot 2^{-y_2}+A_2$$

符号列C_{i-1}及び被保持小数成分x_{i-1}からi番目の情報源シンボルa_iを符号化する場合の式は、

$$L_i = \ell_i + x_{i-1} = y_i + x_i$$

$$C_i = A_i + 2^{-y_i} C_{i-1}$$

となる。被保持小数成分x(s)を有する符号列C(s)として既に符号化されている情報源シンボル列sへ情報源シンボルa_kを付加した場合について上式を書き直すと次のようになる。

$$L(sa_k) = \ell(a_k) + x(s) = y(sa_k) + x(sa_k)$$

$$C(sa_k) = A_k + C(s) \cdot 2^{-y(sa_k)}$$

A_k及び"r"についての要件

ここで、復号を可能にする上で重要な事柄について説明する。各シンボルa_kには、符号化時のC(s)のシフト量に影響を及ぼす長さ$\ell(a_k)$が対応している。このシフト量に影響を及ぼす他の要素は、前の被保持小数成分x(s)である。シフト量$y(sa_k)$、新しい被保持小数成分$x(sa_k)$及び被加数A_kは、算術符号化不等式と呼ばれる次の不等式を満たす。

$$A_{k+1} > A_k + 2^{-y(sa_k)} \cdot C(s)$$

言い換えれば、隣り合うシンボルa_k及びa_{k+1}に各々対応する被加数A_k及びA_{k+1}は、適切な大きさだけ異なっていなければならず、更にシフト量は、$2^{-y(sa_k)} \cdot C(s)$を十分に小さくして、和がA_{k+1}よりも小さくなるように、十分大きくなければならない。もし上述の条件が満たされなければ、算術符号化を首尾よく行なうことはできない。$C(sa_k)$を復号する際には、$C(sa_k)$はA_k以上であることがわかつている(A_kは被加数だがら)。また、上述の不等式から、$C(sa_k)$がA_{k+1}よりも小さいこともわかつており、更に$C(sa_k)$のオペランド部分がrビットであること及び$x(sa_k)$がqビットであることもわかつている。従つて、$C(sa_k)$及び$x(sa_k)$の可能性のある組合わせの数は2^{r+q}である。各々の組合わせについて、$C(sa_k)$を生ぜしめたシンボルa_kが予め決められ、そしてその序数即ち位置番号kが2^{r+q}ワードのメモリの対応するワード位置に記憶される。

次に、符号化及び復号を可能にするA_kのテーブルについて考える。上述の算術符号化不等式から、

$$A_{k+1} - A_k > C(s) \cdot 2^{-y(sa_k)}$$

であることがわかる。

実際には、A_kの値はrビットでの近似値である。被加数A_kの精度は、C(s)の達成可能な最大値によつて左右される。これは、先行の被保持小数成分の値x′及び前に符号化されたシンボルのインデックスk′によつて異なる。現在の、被保持小数値をxで表わし、前の値をx′で表わすと、$y(sa_k) + x = \ell(a_k) + x'$であるから、$y(sa_k) = \ell(a_k) + x' - x$となる。これから、x及びkの与えられた値について、x′が(1＋x－$\ell(a_k)$の小数成分)の小数成分に等しいことがわかる。$A_N(x) > A_{N-1}(x) > \cdots > A_1(x)$であるから、与えられた被保持小数値xに対して、$max\,C_x > A_N(x)$である。ここで、$max\,C_x$は、現在の被保持小数値xを用いて生成(符号化)され得るC(sa_k)の最大値を表わしている。従つて、復号の基準は、$\ell(a_k)$をℓ_kで表わすと、k＜Nなる各k及び可能な小数値xに対して、

$$A_{k+1}(x) - A_k(x) > max\,C_{x'} \cdot 2^{\ell_k - x' + x}$$

である。この不等式は、$A_N(x)$について直接の制限を加えるものではない。しかしながら、もし$A_N(x')$が大き過ぎると、$max\,C_x$も大きくなつて、$A_{k+1}(x) - A_k(x)$を大きくし、更にこれにより$A_N(x)$を大きくする。この問題を解決するためには、$max\,C_x$及び$A_N(x)$の上限を定めなければならない。すべてのxについて、もし$A_N(x) \cdot 2^{-x} + 2^{-\ell_N} \leq 1$であれば、即ち、$A_N$

$(x)+2^{x-\ell_N} \leq 2^x$ であれば、$max C_x \leq 2^x$ が成立する。

Cが最大であつて、シンボル a_N が符号化される場合には、

$$max C_x = A_N(x) + max C_{x'} \cdot 2^{-\ell_N - x' + x}$$

となり、これの両辺に 2^{-x} を乗じると次式が得られる。

$$max C_x \cdot 2^{-x} = A_N(x) \cdot 2^{-x} + max C_{x'} \cdot 2^{-x'} \cdot 2^{-\ell_N}$$

上式から、すべてのxについて、もし $max C_x = 2^x$ であれば、$A_N(x) \cdot 2^{-x} + 2^{-\ell_N} = 1$ となる。逆も同様である。また、すべてのxについて、もし $A_N(x) \cdot 2^{-x} + 2^{-\ell_N} < 1$ であれば、$max C_x < 2^x$ が成立することも導ける。$A_k(x)$ の値の決定については、あとの実例のところで説明する。

復号アルゴリズム

第3図に示した算術列復号装置はバッファ／コントローラ19からパス21を介して圧縮された符号化列を受取り、そして復号されたシンボルをパス25を介してバッファ／コントローラ19へ送る。適切な圧縮されたビット列セグメントをシフトさせるのに必要な長さ情報は、パス22の一部である線123を介して送られる。

復号装置を初期設定するため、最後の被保持小数値 $x(s a_k)$ がバッファ／コントローラ19からパス22の線101を介してXレジスタ103へロードされる。$C(s a_k)$ の上位のrビットは、Cレジスタ95に置かれる。Cレジスタ95からのrビットは、Xレジスタ103からのqビットと共に、復号されるべき最終シンボル a_k の位置番号kをメモリ99から得るためのアドレスを構成する。A_k の値は、メモリ99から線107へ取出されたkを表わすnビット及び線105を介して供給される $x(s a_k)$ を表わすqビットでメモリ109をアクセスすることにより得られる。

符号化された要素を抽出するためには、圧縮された最後の符号列 $C(s a_k)$ を受取るだけでよく、先行の符号列を受取つて保持しておく必要はない。先行の符号列、例えばC(s)は、次のようにして得られる。まず、パス97を介して得られる符号列 $C(s a_k)$ と、メモリ109からパス111へ既出された A_k との差が、減算器113で計算され、この結果、$y(s a_k)$ ビットだけ右シフトされたC(s)即ち $2^{-y(s a_k)} \cdot C(s)$ がパス115へ出力される　パス115上のシフトされたC(s)は、符号化されたときの形へ戻すために、左シフト装置91へ供給される。このとき、バッファ／コントローラ19は、左シフト装置91へ"充填"ビット即ち符号化のときにバッファ／コントローラ5へシフトされた下位の $y(s a_k)$ ビットを供給する。左シフト装置91は、パス123から入力される $y(s a_k)$ に応答して、$y(s a_k)$ ビットだけ左シフトを行ない、これによりパス93上にC(s)が得られる。

$\ell(a_k)$ の整数成分 $y(a_k)$ 及び小数成分 $x(a_k)$ は、パス107を介してメモリ117をアクセスすることによつて得られる。図示のように、$y(a_k)$ はパス119を介して＋1加算器121へ入力され、$x(a_k)$ は減算器125の減数入力へ供給される。減算器125の被減数入力には、小数成分 $x(s a_k)$ が供給される。減算器125は、$x(s a_k) - x(a_k)$ を計算して、パス129へ小数成分x(s)を出力し、更に、もしあれば反転器127へ借り信号を送る。新しい符号ワードC $(s a_k)$ の長さ表示 $L(s a_k)$ の整数成分 $y(s a_k)$ は、反転器127からの借り信号の補数と $y(a_k)$ とを加算器121で加算することによつて得られる（＋1加算）。

最終シンボルの復号時には、レジスタ95は符号列Cの初期値（普通は0）を、そしてレジスタ103は被保持小数成分の初期値を各々含んでいるはずである。これは、符号列の記憶及び伝送の検査に利用できる。

a_k についての実際のシンボル値は、kをパス105から変換器131へ供給することによつて得られる。この変換操作は、第2図に示した符号

装置における変換器５３の操作とは逆になつている。変換器１３１の出力は、バス２５を介してバッファ／コントローラ１９へ送られる。

最後に、これまで説明してきた算術符号化法を実際に適用した場合について述べる。

実例

1.被加数A_kの決定

情報源アルファベットSは４つのシンボルa_1、a_2、a_3及びa_4から成つており、各々の生起確率を$p_1=0.65$、$p_2=0.075$、$p_3=0.025$、$p_4=0.25$とする。このような情報源Sにおいては、エントロピーHは、１シンボル当り1.317729ビットであり、対応するハフマン符号の長さは、$\ell_1=1$、$\ell_2=3$及び$\ell_4=2$である。シンボル当りの平均長は1.45ビットであり、これは、情報源Sのシンボルを２ビットとすると、即ち、$a_1=00$、$a_2=01$、$a_3=10$及び$a_4=11$とすると、平均長が情報源シンボルの長さの72.5%になる符号列を与える。理論上の最良の圧縮率は約65.86%即ち1.317729ビットである。

長さ表示に１つの小数ビットを用いる（$q=1$）算術符号化を考える。ここでの問題は、１つの２進小数ビットを有する有理数であつて、クラフトの不等式を満たしながら平均長を最小にする一組の長さℓ_1、ℓ_2、ℓ_3及びℓ_4を見つけることである（$\ell_k\times 2^1$は整数になる）。この場合、$\ell_{\log_2}(1/p_i)$から理想長を計算すると、0.621449、3.736997、5.321932及び2となる。エントロピーは、これらの理想長から計算することができる。これらの値から長さを求めると、$\ell_1=0.5$、$\ell_2=4$、$\ell_3=4.5$、及び$\ell_2=2.5$となり、これらをクラフトの不等式の左辺に代入すると0.990578となつて、１よりも小さい。これらの長さから計算される平均長は1.3625であつて、68.125%の圧縮率が得られ、これは72.5%の圧縮率を与える上述の単一シンボル・ハフマン符号化よりも優れている。圧縮率を改善するため、ハフマン符号においては、一時に２個のシンボルをブロック化して、情報源符号アルファベットの要素を１６個に増やすことが必要である。これを行なうと、圧縮率は66.76%になる。

$\ell_1=0.5$、$\ell_2=4$、$\ell_3=4.5$及び$\ell_4=2.5$についてA_kの値を求める場合には、これらの長さを与える理想確率$p'_k=2^{-\ell_k}$を考えればよい。即ち、$p'_1=0.7071$、$p'_2=0.0625$、$p'_3=0.044194$及び$p'_4=0.176778$である。

小数成分xが0の場合には、$A_k(0)$の値は、理想累積確率P'_{k-1}の近似値（小数ビットはq個）を表わす。最小の確率p'_3を２進表示すると、0.0000101101…であるから、少なくとも５個の小数ビットが必要である。

$$A'_1(0)=P'_0=0 \qquad =0.00\cdots$$
$$A'_2(0)=P'_1=0.7071067+=0.10110101000+$$
$$A'_3(0)=P'_2=0.7696067+=0.11000101000+$$
$$A'_4(0)=P'_3=0.8138009+=0.11010000010+$$

次になすべきことは、復号可能性の基準を満たすように、$A'_k(0)$をｒビット（１整数ビット及びｒ－１個の小数ビット）で近似することである。$\max C_0<2^0$及び$\max C_{0.5}<2^{0.5}$が満足されているものとする。ｒ－１（＝５）個の小数ビットに丸めると次のようになる。

第1試行（x＝0）	$A_{k+1}-A_k$	x'	$2^{x'}\cdot 2^{-\ell_k-x'+x}$
$A_1(0)=0.0$	0.71825	0.5	0.707106781 2
$A_2(0)=(0.10111)_2=0.71825$	0.0625	0	0.0625
$A_3(0)=(0.11001)_2=0.78125$	0.0625	0.5	0.044194174
$A_4(0)=(0.11011)_2=0.84375$		0.5	0.176776695 3

maxC$_0$＝1及びmaxC$_{0.5}$＝1.11414とすると、k＝1、2及び3に対しては、$A_{k+1}(0)-A_k(0)>\max C_{x'}\cdot 2^{\ell_k+x'}$が成立する。k＝4に対しては、$A_4(0)-2^{-\ell_4}=1.020526695>1$となるが、これは、小数ビットの数（r－1）を6にして、第1試行における$A_2(0)$、$A_3(0)$及び$A_4(0)$から2^{-6}を引くことによつて補正することができる。

第2試行（x＝0）	$A_{k+1}-A_k$	$2^{x'}\cdot 2^{-\ell_k-x'+x}$
$A_1(0)=0.0$	0.703125	0.707106781 2
$A_2(0)=(0.1011101)_2=0.703125$	0.0625	0.0625
$A_3(0)=(0.1110001)_2=0.765625$	0.0625	0.044194174
$A_4(0)=(0.1101101)_2=0.828125$		0.176776695 3

この場合、$A_4(0)-A_3(0)=0.0625$となつて、$2^{-\ell_3}=0.044194174$との差が2^{-6}よりも大きく、従つてA_4を2^{-6}だけ小さくすることができる。

第1試行（x＝0.5）	$A_{k+1}-A_k$	x'	$2^{x'}\cdot 2^{-l_k-x'+x}$
$A_1(0.5)=0.000000=0$	1	0	1
$A_2(0.5)=1.000000=1$	0.09375	0.5	0.08838834765
$A_3(0.5)=1.000110=1.09375$	0.0625	0	0.0625
$A_4(0.5)=1.001010=1.15625$		0	0.25

$A_4(0.5)$を調べてみると、$A_4(0.5)+2^{0.5-l_4}=1.406625$であるから、$2^{0.5}=1.414214$よりも小さい。$\max C_0$及び$\max C_{0.5}$は、シンボル$a_4$の列を符号化することによって決定できる。Cは0に初期設定され、C_0は2^0だけ大きくされ$C_{0.5}$は$2^{0.5}$だけ大きくされる。これらの単調増加シーケンスは$\max C_0$及び$\max C_{0.5}$に各々収束する。$\max C_0$は0.9879032758（2進表示では、0.111111100111100111001111・・・0011111・・・）に決定された。$\max C_{0.5}$は1.4032258 06（2進表示では1.0110011110011110011111・・・）である。$\max C_{0.5}$のこの新しい値を用いると、$A_2(0)-A_1(0)=0.703125$と$\max C_{0.5}\cdot 2^{-l_1-0.5}=1.403225806\times 0.5=0.701612903$との比較から、復号可能性の基準が満たされていることがわかる。

もし復号可能性の基準が満たされていなければ、理想被加数$A'_k(x)$をより高い精度で近似するよ

この場合は、$A_4+2^{-l_4}=0.8125+2^{-2.5}=0.989276695 3<2^0=1$となるが、$A_2(0)-A_1(0)$が$2^{-l_1}$よりも小さくなるという問題がある。しかしながら、もし$\max C_{0.5}$が十分に小さければ、復号可能性の基準は、$\max C_0$（$<2^0$）及び$\max C_{0.5}$（$<2^{0.5}$）の実際の値に基づいて満たされ得る。

$x=0$の場合と同じようにして、$x=0.5$のときの理想被加数$A'_k(0.5)=2^{0.5}\cdot P'_{k-1}$を求めてみる。

第3試行（x＝0）	$A_{k+1}-A_k$	$2^{x'}\cdot 2^{-l_k-x'+x}$
$A_1(0)=0.0$	0.703125	0.7071067812
$A_2(0)=(0.101101)_2=0.703125$	0.0625	0.0625
$A_3(0)=(0.110001)_2=0.765625$	0.046875	0.044194174
$A_4(0)=(0.110100)_2=0.8125$		0.1767766953

$A'_1(0.5)=0.0$ $=0$

$A'_2(0.5)=1$ $=(1.00000000)_2$

$A'_3(0.5)=1.088388348=1.000101110+$

$A'_4(0.5)=1.150888348=1.001001110+$

上述の理想値を丸めると次のようになる。

うに、小数ビットの数（ｒ－１）を増やす必要がある。$\Sigma 2^{-\ell_k} < 1$ である限り、$q > 0$（長さ ℓ_k が非整数即ち小数）についての解は常に存在する。

符号化装置９及び復号装置２３のメモリはロードされねばならない。符号化用メモリ５７の内容は次の通りである。なお、添字の" ２ "は２進表示を表わしている。

アドレス(k)	ℓ_k
００	$(0.0001)_2$
０１	$(0.100)_2$
１０	$(0.1001)_2$
１１	$(0.101)_2$

メモリ５９の内容は次の通りである。

アドレス（ｘ、ｋ）	$A_k(x)$
０、００	$(0.0000000)_2$
０、０１	$(0.1011101)_2$
０、１０	$(0.1100001)_2$
０、１１	$(0.1101001)_2$
１、００	$(0.0000000)_2$
１、０１	$(1.0000000)_2$
１、１０	$(1.0001110)_2$
１、１１	$(1.0010101)_2$

復号装置２３においては、メモリ１０９及び１１７の内容は、符号化装置９のメモリ５９及び５７の内容と各々同じであるが、メモリ９９は次のようになつている。

アドレス（ｘ、C_x）	ｋについての符号化	シンボル
０、0.0000000	００	a_1
０、0.1011100	００	
０、0.1011101	０１	a_2
０、0.1100000	０１	
０、0.1100001	１０	a_3
０、0.1100011	１０	
０、0.1101001	１１	a_4
０、1.1111111	１１	
１、0.0000000	００	a_1
１、0.1111111	００	
１、1.0000000	０１	a_2
１、1.0001010	０１	
１、1.0001110	１０	a_3
１、1.0010010	１０	
１、1.0010100	１１	a_4
１、1.1111111	１１	

このメモリは、C_x（及び被保持小数値ｘ）の値と $A_k(x)$ との同時比較を可能にする。

2.シンボル列の符号化

シンボル列 $a_4 a_2 a_1 a_3$ が符号化されるものとする。Ｃレジスタ８７の内容をＣで表わし、ｉ番目の符号化ステップにおける符号列の全体を $C^{(i)}$ で表わす。符号化ステップは次の通りである。

刻時１

$a_4 \rightarrow \ell_4 = 2.5$、$x = 0.5$、$C^{(1)} = A_4(0.5) = (1.0010100)_2$

刻時２

(a) $a_2 \rightarrow \ell_2 = 4$、$y = 4$、$x = 0.5$、$A_2 = (0.5) = (1.0000000)_2$

(b) $C^{(1)}$ が４ビットだけ右シフトされて、$A_2(0.5)$ に加算される：$C^{(2)} = 1.0001001 0100$。下位の４ビットは、バッファ／コントローラ５へ送られる：$C = 1.0001000$。

刻時3

(a) $a_{,1} \rightarrow \ell_1 = 0.5$、$y = 1$、$x = 0$、$A_1(0) = 0$

(b) Cは1ビットだけ右シフトされて（最下位ビットはバッファ／コントローラ5へ送られる）、$A_1(0)$に加算される：$C^{(3)} = 0.1000100$ 1010、$C = 0.100010$。

刻時4

(a) $a_3 \rightarrow \ell_3 = 4.5$、$y = 4$、$x = 0.5$、$A_3(0.5) = 1.0001110$

(b) $C^{(3)}$は4ビットだけ右シフトされて（下位の4ビットはバッファ／コントローラ5へ送られる）、$A_3(0.5)$に加算される：$C^{(4)} = 1.0010000001001010$、$C = 1.0010000$。

3. 復号

この場合の符号列はC＝1.0010000001

00101000であり、被保持小数成分は$x = 0.5$である。復号装置23のXレジスタ103は$x = 0$に初期設定され、7ビットのCレジスタ95には1.0010000が与えられる。

刻時1

(a)メモリ99へアドレス（x、C）＝（1、1.0010000）を供給して、$k_{00} = a_3$を取出す。

(b) $\ell_3 = 4.5$、新しい$x = 0$、$y = \ell_3$＋新x－被保持小数値＝4

(c) 1.0010000から$A_3(0.5) = 1.0001110$が引かれる→0.0000010

(d)この減算結果が4ビットだけ左シフトされ（→0.10）、バッファ／コントローラ19からの次の4ビット（0010）がCレジスタ95に記憶される：0.1000010。（バッファ内のC＝0101 0）

刻時2

(a)メモリ99へアドレス（x_1、C）＝（0、0.1000010）を供給して、k（00）＝a_1を取出す。

(b) $\ell_1 = 0.5$、新$x = 0.5$、$y = 0.5 + 0.5 = 1$

(c)＝0.1000010から$A_1(0) = 0.0000000$が引かれる。

(d)減算結果が1ビットだけ左シフトされ、バッファ19から1つの0ビットが供給される：C＝1.0000100、バッファ内のC＝1010

刻時3

(a)メモリ99へアドレス（x、C）＝（1、1.0001000）を供給して、k（01）＝a_2を取出す。

(b) $\ell_2 = 4$、新$x = 0.5$、$y = 4.5 - 0.5 = 4$

(c)＝1.0001000から$A_2(0.5) = 1.0000000$が引かれる→0.0001000

(d)減算結果が4ビットだけ左シフトされ、空いた所にバッファ／コントローラ19からの4ビット（1010）が供給される：C＝1.0010100、バッファ／コントローラ19は空

刻時4（バッファ／コントローラ19は空であつて、最終ステップであることを知らせる）

(a)メモリ99へアドレス（x、C）＝（1、1.0010100）を供給して、$k_{00} = a_4$を取出す。

(b) $\ell_4 = 2.5$、新$x = 0$、$y = 2$

(c) C＝1.0010100から$A_4(0.5) = 1.0010100$が引かれる→0.0000000

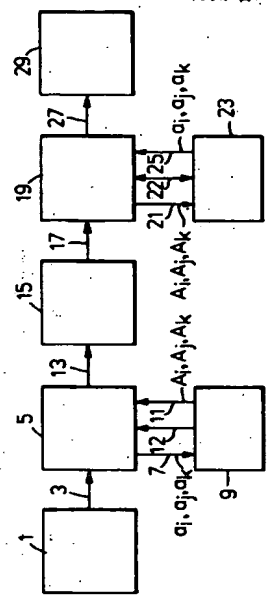
(d)バッファ／コントローラ19はシフト量を無視する。復号完了。

復号が完了すると、Cレジスタ95及びXレジスタ103の内容は0になるが、もし異なつた値が含まれていると、何らかのエラーが生じたことになる。

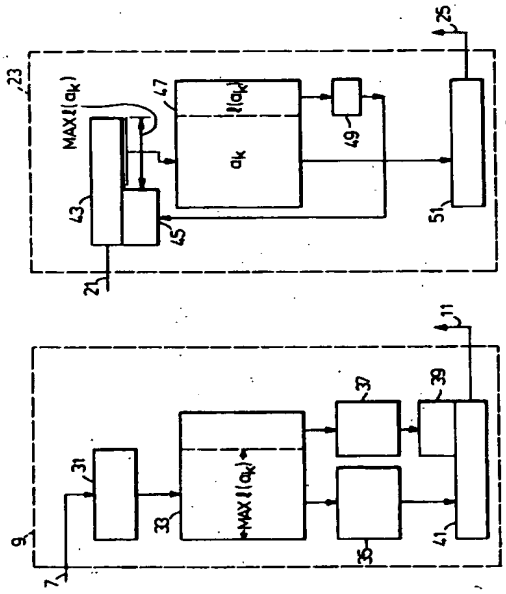
4. 図面の簡単な説明

第１Ａ図は符号化及び復号を行なうシステムの概略を示すブロック図、第１Ｂ図はハフマン符号化装置のブロック図、第１Ｃ図はハフマン復号装置のブロック図、第２図は本発明に従う符号化装置のブロック図、第３図は第２図の符号化装置に関連して使用される復号装置のブロック図、第４Ａ図は第２図の符号化装置のためのタイミング図、第４Ｂ図は第３図の復号装置のためのタイミング図である。

１・・・情報源、５、１９・・・バッファ／コントローラ、９・・・符号化装置、１５・・・ＬＩＦＯ記憶ユニット、２３・・・復号装置、２９・・・利用装置、３１・・・入力レジスタ、５３・・・変換器、５７、５９・・・メモリ、６３、８３・・・加算器、６７・・・Ｘレジスタ、７５・・・＋１加算器、７９・・・右シフト装置、８７・・・Ｃレジスタ。



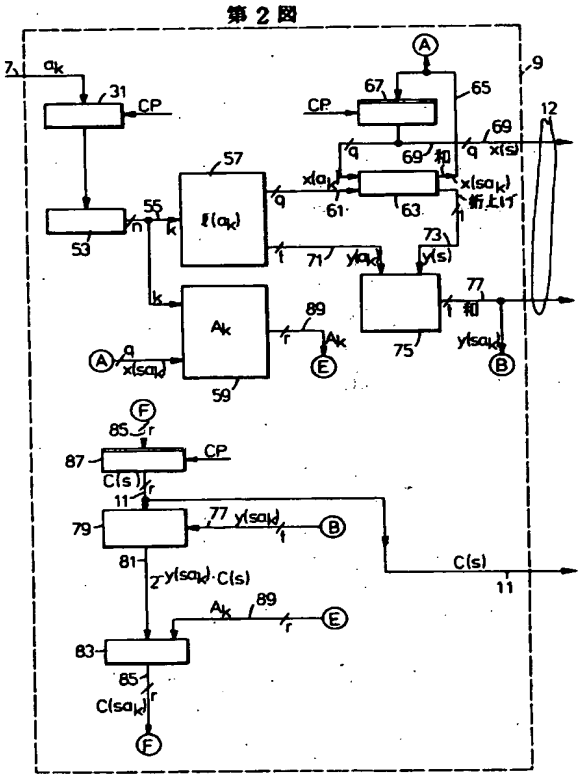
第１Ａ図



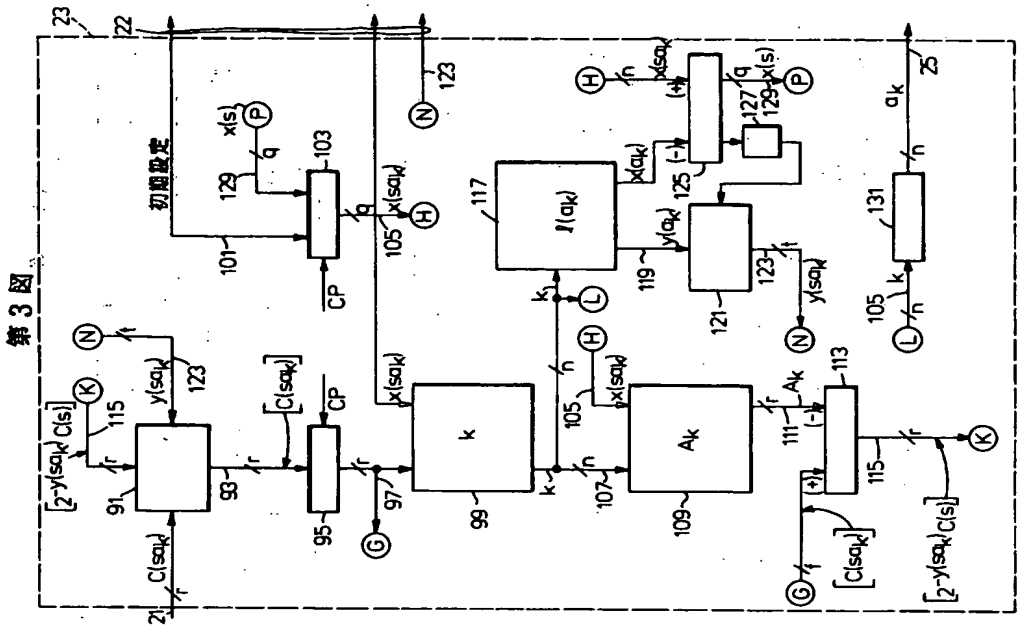
第１Ｃ図

第１Ｂ図

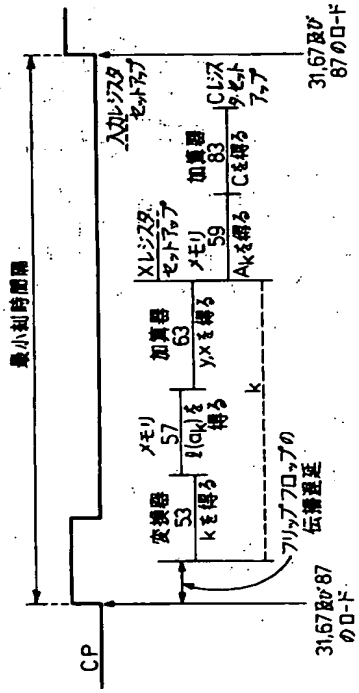
第２図



第3図



第4A図



第4B図